Tampereen yliopisto

Riku Sundqvist

OHJELMISTOPROJEKTIEN EPÄONNISTUMISTEN EHKÄISEMINEN RISKIENHALLINNALLA

Tekniikan ja luonnontieteiden tiedekunta
Kandidaatintyö
Joulukuu 2019

TIIVISTELMÄ

Riku Sundqvist: Ohjelmistoprojektien epäonnistumisten ehkäiseminen riskienhallinnalla / Preventing software project failures with risk management
Kandidaatin työ
Tampereen yliopisto
Tietojohtaminen
Joulukuu 2019

Tutkimusongelmana pidin ohjelmistoprojektien epäonnistumista. Ongelma on hyvin yleinen ja sen seuraukset vakavia. Ohjelmistoala on myös kasvanut merkittävästä ja digitalisaation myötä toimivat ohjelmistot ovat yhä tärkeämpiä niin yksilölle, kuin yhteiskunnalle. Lähestyin ongelmaa tutkimuskysymyksillä ”mitkä tekijät vaikuttavat ohjelmistoprojektien epäonnistumiseen?” ja ”miten epäonnistumisia voisi ehkäistä riskienhallinnalla?”. Valitsin nämä kysymykset, koska epäonnistumisiin johtavien tekijöiden tunnistaminen auttaa tutkimaan, miten epäonnistumisia voisi ehkäistä. Rajasin ehkäisemisen näkökulman riskienhallintaan, koska aihe kiinnostaa minua ja uskon riskienhallinnan hyödyntämisellä olevan suuri merkitys projektin onnistumiselle.

Etsin epäonnistumisen syitä erityisesti projektinhallinnan näkökulmasta, koska riskienhallinta on osa projektinhallintaa. Kokonaisuuden hahmottamiseksi käytin lähdeaineistona tieteellisiä julkaisuja, joissa aihetta on tutkittu erilaisista näkökulmista ja tehty päätelmiä erilaisin perustein. Samalla tavalla tarkastelin riskienhallintaa löytäen sekä aiheen yleispiirteitä, että erilaisia tulkintoja riskeistä ja riskienhallinnan toteuttamisesta ja sen haasteista. Käsitteistä määritin vain tutkimusongelmani- ja kysymysteni olennaisimmat.

Vaikka tutkimusaineistosta tunnistin useita projektinhallinnan haasteita, riskejä ja riskienhallinnan näkökulmia, lopulta ne pohjautuivat vain pariin merkittävään ongelmaan. Projektin osapuolten välisen kommunikaation ongelmat johtavat väärinkäsityksiin ja epävarmuuteen, mikä lisää projektin riskialttiutta. Väärinkäsitykset ja epävarmuus altistavat projektin yllättäville muutoksille ja riskeille, joihin ei ole varauduttu, mikä taas johtaa usein projektin budjetin ylittymiseen, projektin myöhästymiseen tai sen kokonaan perumiseen. Toinen merkittävänä havaittu ongelma oli projektien monimuotoisuus, mikä vaikeuttaa projektin johtamista ja riskienhallinnan soveltamista.

SISÄLLYSLUETTELO

1. JOHDANTO 1
2. TUTKIMUKSEN AIHE JA TUTKIMUSMENETELMÄT 3
2.1 Tutkimusongelma, -kysymys, -menetelmä ja lähdeaineiston haku 3
2.2 Tutkimusaineisto 5
3. OHJELMISTOPROJEKTIT -JA KEHITYSMALLIT 7
3.1 Keskeisiä käsitteitä 7
3.2 Ohjelmistoprojektienhallinta ja sen haasteet 8
3.3 Näkökulmia ohjelmistoprojekteihin 11
4. RISKIT OHJELMISTOPROJEKTEISSA 15
4.1 Riskien jaottelu 15
4.2 Riskien yleisyys 18
4.3 Riskienhallintaprosessimalleja 19
4.4 Riskienhallinnan haasteita 20
5. OHJELMISTOPROJEKTIEN EPÄONNISTUMISTEN ENNALTAEHKÄISY RISKIENHALLINNALLA 22
5.1 Riskienhallinnan hyödyntäminen projektinhallinnassa epäonnistumisten ehkäisemiseksi ja haasteiden minimoimiseksi 22
5.2 Haasteet yhteensovittaessa riskienhallinnan menetelmiä projektinhallintaan 24
6. YHTEENVETO 26
LÄHTEET 28

1. JOHDANTO

Ohjelmistoprojektit ja niihin liittyvä liiketoiminta ovat yhä merkittävämpi osa jatkuvasti digitalisoituvaa maailmaa. Ne ovat ohjelmistoyritysten ydinliiketoimintaa, joten myös toimitettujen projektien tulee olla onnistuneita ja toimivia asiakasorganisaatiossa, jotta asiakassuhteiden ja liiketoiminnan jatkuvuudelle on edellytykset. Syyt ohjelmistoprojektien epäonnistumiseen syntynevät projektin aikana jo varhain ennen kuin ne huomataan.

Minua kiinnosti tietää, miten ohjelmistoalan projektien riskejä voi tunnistaa ja miten varautua niihin etukäteen, koska saatan työskennellä tulevaisuudessa projektipäällikkönä, konsulttina tai muuten ohjelmistotuotannossa. Näissä tehtävissä uskon sekä projektinhallinnan haasteiden, että riskienhallinnan tuntemisen auttavan suuresti. Täten aiheen tutkiminen kiinnosti minua myös ammatillisen sivistämisen näkökulmasta. Uskon voivani hyödyntää tutkimuksesta oppimaani myös maisterivaiheen opinnoissani.

Ohjelmistoala on jatkuvasti kasvanut ja siten projekteille on yhä enemmän kysyntää, projektien rahasummat yhä suurempia sekä projektit yhä mittavampia. On arvioitu, että vuonna 1985 ohjelmistokauppojen yhteissumma oli 140 miljardia euroa, vuonna 1995 summa oli jo 450 miljardia euroa, kun taas vuonna 2005 arvioitiin summan olevan jopa 1400 miljardia euroa. Voidaan siis päätellä ohjelmistoalan olevan todella merkittävä, koska sen kasvu on ollut nopeaa. Projektien ollessa yhä suurempia ja alan teknologioiden ja innovaatioiden kehittyessä on selvää, että ohessa myös haasteet ja riskit kasvavat ja saavat uusia muotoja. (McManus 2012, s. 1)

Viime vuosikymmenen jälkimmäisellä puoliskolla ohjelmistoprojekteista epäonnistui tai peruutettiin täysin yli kymmenesosa (Emam & Koru 2008, s. 87). Ohjelmistoprojekteja epäonnistuu siitä huolimatta, että ajan kuluessa on kehitetty esimerkiksi ketteriä ohjelmistotuotantomenetelmiä, joissa jatkuva ohjelmiston testaus ja palautteenanto ovat olennainen osa projektia koko sen elinkaaren ajan (Coleman 2016, s. 23). Etsimällä projekteja yhdistävät virheet ja riskit pyrin tunnistamaan systemaattiset virheet, joita epäonnistuneissa projekteissa ilmenee.

Varsinkin yhteyskunnalliset julkisin varoin tilatut projektit lienevät kalliita ja niiden vaikutukset merkittäviä. Tällä vuosikymmenellä Suomessa muun muassa VR:n lippujärjestelmän käyttöönotto ja joissakin kunnissa vuoden 2012 kunnallisvaalien sähköinen äänestysjärjestelmä eivät toimineet (Vanhala 2012). Edellä mainitussa junalippujen myynti lamaantui kymmeniksi tunneiksi ja jälkimmäisessä äänestysjärjestelmä hukkasi 232 ääntä. Tältä vuodelta esimerkkinä on hätäkeskuksen uuden hälytysjärjestelmän toimimattomuus (Kriikku 2019). Tällaisten kalliiden ja yhteiskunnallisesti merkittävien ohjelmistoprojektien epäonnistumisten ehkäisemiseksi mielestäni oli tärkeää selvittää, miten ongelmilta voidaan välttyä mahdollisimman hyvin.

Edellä mainituissa tapauksissa järjestelmän viallisuus huomattiin vasta sen oltua otettu käyttöön. Olisiko järjestelmien vikoja ja niihin johtaneita riskejä voinut havaita ja tunnistaa vuosia kestäneiden ja useita miljoonia euroja maksaneiden projektien jo kauan ennen kuin ne kävivät ilmi?

2. TUTKIMUKSEN AIHE JA TUTKIMUSMENETELMÄT

Tässä kappaleessa esittelen ensin tutkimusongelman- ja kysymyksen. Sen jälkeen kerrotaan käyttämästäni tutkimusmenetelmästä tässä kirjallisuustutkimuksessa, sekä esitellään käyttämästäni aineistosta aiheen kannalta keskeisimpänä pitämäni lähteet. Nämä tiedot antavat lukijalle viitekehyksen varsinaisen aiheen käsittelemiselle.

2.1 Tutkimusongelma, -kysymys, -menetelmä ja lähdeaineiston haku

Tutkimusongelmani oli **ohjelmistoprojektien epäonnistuminen**. Tutkimusongelmaani hain vastausta tutkimuskysymyksilläni:

1. Mitkä tekijät vaikuttavat ohjelmistoprojektien epäonnistumiseen?
2. miten epäonnistumisia voisi ehkäistä riskienhallinnalla?

Tämän tutkimuksen suoritin kirjallisuustutkimuksena. Kirjallisuustutkimuksessa analysoidaan aihetta ainoastaan lähdekirjallisuuden avulla. Tavoitteena on vertailla lähteitä esittämiä näkökulmia aiheeseen ja saada ne ikään kuin ”keskustelemaan” keskenään. (Vaismaa 2009, s. 8)

Vastatakseni näihin kysymyksiin etsin lähteistä epäonnistumisten syitä ja riskejä, jotka voivat johtaa projektin epäonnistumiseen. Lisäksi tarkastelin, onko joitain yhdistäviä tekijöitä epäonnistuneiden projektien välillä tai jotain tunnistettavaa ja toistuvaa kaavaa, joka johtaa virheisiin. Etsin aluksi kustakin aihepiiristä tietoa eri lähteistä ja erilaisista näkökulmista. Tämän jälkeen toin ilmi aineistossa esitettyjä näkemyksiä ja toteamuksia, sekä tulkitsin ja vertailin niitä.

Etsin tietoa ensisijaisesti tieteellisistä julkaisuista. Hyödynsin paljon yliopiston hakukoneita, kuten Andoria ja Tunilibia. Tutkimussuunnitelmaa tehdessäni huomasin, että lopulta päädyn helposti muihin tieteellisten teosten hakukoneisiin, kuten ProQuest ja Web of Science. Havaitsin myös, että Google Scholarista löytyy paljon lähteitä hyvinkin tarkoilla hakusanoilla. Tosin monesta hakutuloksesta aineisto ei ollut saatavilla ilmaiseksi tai edes sähköisessä muodossa.

Hakusanoja, joista aloitin, olivat muun muassa “software project failure”, “software project risks”, “software project management”, “Software project risk”. Hakutuloksiin vaikutti myös, käyttikö ”Software” vai ”IT” sanaa. Molemmilla löytyi hyödyllisiä lähteitä. Huomasin jo aiheanalyysivaiheessa, että lähteitä kannattaa hakea suoraan englanniksi, koska aineistoa löytyi silloin niin paljon enemmän ja tarkemmilla hakusanayhdistelmillä, kuin suomeksi. Suomeksikin aioin varmuuden vuoksi yrittää, esimerkiksi hakusanoin ”ohjelmistoprojekti vaiheet” ja ”ohjelmisto epäonnistuminen”, mutta hyödyllisiä lähteitä en löytänyt. Hakutulosten määrä eri hakusanoilla Andorissa ja Google Scholarissa on koottu taulukkoon 2.1.

Hakusana	Andor	Google Scholar
IT project management	13 771 124	~ 6 310 000
IT project risk	9 757 409	~ 5 350 000
IT project failure	3 742 155	~ 4 800 000
Software project management	2 731 226	~ 4 110 000
Software project risks	1 651 772	~ 3 270 000
Software project risk management	1 215 649	~3 010 000
Software project failure	674 228	~ 3 200 000

Taulukko 2.1: Käyttämiäni hakusanoja ja niitä vastaavat hakutulosten määrät

Suunnittelin myös hyödyntäväni uutisartikkeleita tilanteissa, joissa ne olisivat tukeneet käytännön esimerkein tieteellisistä julkaisuista tekemiäni havaintoja ja väittämiä. Lopulta käytin niitä vain johdannossa, koska sopivia kohtia hyödyntää niitä en tutkimuksen aihetta käsittelevissä kappaleissa todennut. En myöskään kokenut arvokkaana käyttää aikaa uutisartikkeleiden etsimiseen, koska sain enemmän hyötyä tutkimukseeni tieteellisistä teksteistä.

Vaikeuksia tuotti niin epäonnistuneissa, kuin onnistuneissa projekteissa yhdistävien tekijöiden havaitsemisen vaikeus, koska ne eivät välttämättä ole selkeitä jokaisen projektin ollessa uniikki. Myöskään onnistuneen projektin määritelmä ei ole yksiselitteinen, koska usein projekti ei ole joko täysin epäonnistunut tai täysin onnistunut, vaan jotain niiden väliltä (Emam & Koru 2008).

Tässä tutkimuksessa pidin epäonnistunutta projektia sellaisena, joka ei täytä sille asetettuja vaatimuksia. Epäonnistumisena pidin myös sitä, että projekti on myöhässä suunnitellusta aikataulusta tai ylittää sille asetetun budjetin. Usein kuitenkaan käyttämissäni lähteissä ei tarkennettu epäonnistuneen projektin määritelmää, vaikka ne käsittelivät projektien epäonnistumista.

2.2 Tutkimusaineisto

Emam & Goru (2008) olivat keränneet tilastoja ja tunnuslukuja kyselynä suorittamaansa tutkimukseen epäonnistuneista ohjelmistoprojekteista. Tutkimus tarjosi riittävän tuoretta tietoa tutkimusongelman havainnollistamiseksi tunnusluvuin. Vaikka tutkimus oli yli kymmenen vuotta vanha, siinä todetut ongelmat olivat yhteneväisiä tuoreempien tutkimusten kanssa.

Kumar & Rashid (2018) tarkastelivat ohjelmistojen elinkaarta eri vaiheissa sen luomisen ja käytöstä poiston välillä. He myös soveltivat aineistoaan erilaisiin ohjelmistokehitysmalleihin, kuten vesiputousmalliin ja spiraalimalliin, ja vertailivat niitä. Tämän lähde auttoi ymmärtämään projektinhallinnan näkökulman ohjelmistoprojekteissa.

Esimerkiksi Tesch et al. (2007) ja Menezes et al. (2019) tutkivat ohjelmistoprojekteja riskienhallinnan näkökulmasta. Muun muassa nämä olivat tärkeimpiä lähteitäni, koska tutkimuskysymykseni löysin paljon hyödyllisiä huomioita. Varsinkin jälkimmäinen lähteistä oli todella hyödyllinen uutuusarvonsa vuoksi ja sen takia myös relevantimpikin, kuin edellä mainittu, mutta Tesch et al. (2007) huomioi näkökulmia, joita Menezesin et al. (2019) tutkimus ei käsitellyt.

Samantra et al. (2016) tarjosi myös hyvin tuoreena pitämääni tietoa. Tutkimus käsitteli riskienhallintaa ja keskittyi riskientunnistusmallin luomiseen. He olivat selvittäneet ohjelmistoprojekteille jopa 23 eri riskiä jaettuna neljään eri ryhmään. Riskien jaon tarkastelu antoi lisää tietoa riskienhallinnan käsittelyyn ja riskien vertailu muiden tutkijoiden riskien vertailuun toi lisäarvoa kirjallisuustutkimuksen näkökulmasta.

Samoin myös Lu et al. (2013) keskittyi ohjelmistoprojektien riskien tunnistamiseen ja jonkinlaisen mallin luomiseen projektin tukityökaluksi. Heidän tutkimuksessaan painottui muihin lähteisiini verrattuna enemmän tekninen näkökulma, koska he käsittelevät erilaisia matemaattisia algoritmeja ja laskelmia, mutta toisaalta myös hyvää kirjallista pohdintaa ja argumentointia. Sinänsä matemaattinen ja tekninen lähestymistapa aiheeseen

olisi varmasti ollut hyödyllinen, mutta sen käsittely olisi tehnyt tutkimuksestani liian laajan, eikä matemaattinen asiantuntevuuteni olisi välttämättä riittänyt sen tulkitsemiseen.

Riskienhallinnan käsittelyn kannalta hyödyllisimpänä ja mielenkiintoisimpana pitämäni lähteen oli kirjoittanut Taylor et al. (2012). Aineistossa pohdittiin syitä, miksi riskienhallinta on niin vaikea toteuttaa ohjelmistoprojekteissa, vaikka aihetta on tutkittu jo kauan ja erilaisia menetelmiä epäonnistumisten ehkäisemiseksi on ehdotettu. Tutkimuksessa myös viitatattiinn aikaisempiin tutkimuksiin aiheesta, kuten tutkimuksessanikin käyttämäni Martin et al. (2007) ja Howell et al. (2010), mikä auttoi edelleen löytämään aiheeseen liittyviä hyödyllisiä lähteitä.

Graafisesti hyödynsin Ahmedin (2009) ohjelmistoprojektinhallintaa käsittelevän tutkimuksen kuvaajia, koska ne olivat selkeitä ja mielestäni tarpeeksi kattavia. Lisäksi tein itse kaaviot Medvedskan & Berzisan (2015) ja Chemeturin (2013) näkemyksistä projekteihin sisältyvistä vaiheista, koska niitä ei aineistossa valmiina ollut. Lisäksi Lin & Huang (2018) käsittelivät projekteihin sisältyviä prosesseja ja heidän tutkimuksensa sisälsi hyödyllisen kuvan prosessien jaottelusta. Myös Cha & Mayoterna-Sanchez (2019) olivat tehneet havainnollistavan kuvan projektissa tarvittavista taidoista. Riskienhallintaa käsitellessä sopivat lähteet havainnollistaville kuville olivat McManusin (2012) jako riskeistä ja Samantran et al. (2016) riskien tarkempi jako, joka esitti myös niiden vuorovaikutssuhteet.

Ohjelmistoprojektien epäonnistumisten ongelman ratkaisemiseksi etsin siis englanniksi lähteitä yliopiston kirjaston Andorista ja Google Scholarista. Pyrin valitsemaan käyttööni lähteet, jotka olivat mielestäni riittävän tuoreita ja relevantteja. Lähteiden valitsemisessa pidin tärkeänä niiden vastaavuutta mahdollisimman kattavasti tutkimuskysymyksiini ohjelmistoprojektien epäonnistumisista ja riskienhallinnasta.

3. OHJELMISTOPROJEKTIT -JA KEHITYSMALLIT

Tässä opinnäytetyössä näkökulma ohjelmistoprojekteihin ja niiden epäonnistumisiin on projektinhallinnallisesta näkökulmasta, vaikka esimerkiksi Sudhakarin (2012, s. 539) mukaan onnistunut ohjelmistoprojekti koostuu sekä onnistuneesta projektinhallinnasta, että onnistuneesta ohjelmistosta. Jotta voin tarkastella ohjelmistoprojektien epäonnistumisiin johtaneita syitä ja pohtia, miten niihin voisi varautua sekä epäonnistumisia ehkäistä, on hyvä käydä läpi yleisesti ohjelmistoprojekteihin ja projektinhallintaan liittyviä käsitteitä sekä piirteitä. Aluksi käyn läpi mielestäni keskeisimmät käsitteet ja sen jälkeen käsittelen yleisesti ohjelmistoprojekteja ja niiden hallintaa, sekä vertailen erilaisia projektimalleja.

3.1 Keskeisiä käsitteitä

Yleisesti **ohjelmisto** tarkoittaa tietokoneohjelmaa ja siihen liittyvää dokumentointia (Sommerville 2006, s. 6). Ohjelmistot voidaan jakaa useisiin alakäsitteisiin, kuten käyttöjärjestelmä-, tietokanta-, verkkopalvelin-, tietoturva-, ja tiedonvälitysohjelmiin (Chemuturi 2013, s. 7-10). Tässä työssä ohjelmistolla voidaan tarkoittaa pelkkää tietokoneohjelmaa, sovellusta tai kokonaista tietojärjestelmää riippumatta sen alalajista.

Ohjelmistotuotanto tarkoittaa insinöörimäisellä lähestymistavalla toteutettua toimintaa tietokoneohjelman kehittämiseksi (Kumar & Rashid 2018, s. 1), mikä koostuu tietyistä suunnitelmallisesti suoritettavista tehtävistä (Chang et al. 2008, s. 1142). Sen voidaan katsoa enenevissä määrin käsittävän toiminnan ja käytetyn teknologian lisäksi myös sitä toteuttavat ihmiset ja heidän väliset suhteensa (Lenberg et al. 2015). Ohjelmistotuotantoon liittyy vahvasti **ohjelmistoprojektinhallinta**, joka tarkoittaa ohjelmistotuotannon tavoiteltuun tuotokseen pääsemisen edistämistä projektinhallinnan menetelmiä soveltaen (Ahmed 2011).

Ohjelmistoprojekteilla tarkoitetaan erilaisista tunnistettavista vaiheista koostuvia prosesseja. Prosessi käsittää ohjelmiston elinkaaren. Erilaisia projektimalleja ja elinkaaria on useita. (Kumar & Rashid 2018, s. 2-3) Ohjelmistoprojektin tavoitteena on tuottaa ohjelmisto, joka vastaa sille määrittelyssä asetettuja kriteerejä ja joka on tuotettu asetetun aikataulun, vaiheiden ja budjetin mukaisesti (Aladwani 2002, s. 217).

Riskeillä yleensä tarkoitetaan negatiivisia riskejä, eli uhkia, jotka toteutuessaan aiheuttavat yhden tai useamman negatiivisen vaikutuksen projektiin (Menezes 2019, s. 1152).

Sen sijaan positiiviset riskit ovat mahdollisuuksia, jotka toteutuessaan tuovat jotain merkittävää hyötyä (Valtiovarainministeriö 2017, s. 11). Yleisesti riskeillä viitataan kuitenkin negatiivisiin riskeihin, kuten myös lähtökohtaisesti tässä tutkimuksessa.

Riskienhallinta tarkoittaa projektinhallinnan osa-aluetta, joka on suunnitelmallista toimintaa riskien arvioimiseksi ja riskeihin varautumiseen projektin aikana. Pohjimmiltaan kyse on siis tulevaisuuden epävarmuuden ja ennustamisen mahdottomuuden aiheuttamien negatiivisten lieveilmiöiden minimoinnista (Dionne 2013, s. 154). Toisaalta riskeihin lukeutuu edellä määritellysti myös positiiviset riskit, joten voitaneen pitää myös positiivisten riskien maksimoimista osana riskienhallintaa, vaikkei tätä näkökulmaa niin useasti tarkastellakaan ainakaan tätä tutkimusta varten löytämissäni alan julkaisuissa.

Ohjelmistokehitystä ja -projekteja käsiteltäessä tässä opinnäytetyössä **määrittelyllä** tarkoitan ohjelmiston vaatimusten määrittelyä projektin määrittelyvaiheessa. Määrittelyssä dokumentoidaan tuotettavalle ohjelmistolle vaadittavat toiminnot ja lainalaisuudet (Butterfield et al. 2016). Tämä vaihe on tärkeä, koska siinä luodaan perusta projektin myöhempien vaiheiden suunnittelulle. Käsitteen ymmärtäminen on tässä tapauksessa mielestäni tärkeää, koska määrittelyvaihetta käsitellään useasti myöhemmissä kappaleissa.

3.2 Ohjelmistoprojektienhallinta ja sen haasteet

Pressmanin (2005) mukaan ohjelmistoprojektienhallinta käsittää henkilöstön, tuotteen, prosessit ja niistä muodostuvan projektin. Sudhakar (2012, s. 541) käsittää projektinhallinnan koostuvan sen sijaan vain inhimillistä resursseista, jotka ovat ylin johto, projektipäällikkö, tiimin jäsenet, järjestelmäarkkitehdit, käyttäjät, myyjät, toimittajat ja asiakkaat. Nämä resurssit kattavat toisaalta hyvin toimintoja hyvin laajalta alalta projektin myymisestä sen toimittamiseen. Koska projektit ovat näin moninaisia ja teknisesti vaativia, vaaditaan niiden suunnitteluun ja johtamiseen erittäin päteviä asiantuntijoita (Lu et al. 2013). Jotta projekti voidaan toteuttaa ja hallita sitä, on sillä oltava selkeästi määritelty tavoite ja arvolupaus, sekä projektin osapuolien ja sidosryhmien oltava sitoutuneita projektin toteuttamiseen suunnitellusti (Cinis 2015, s. 254).

Tehokasta projektinhallintaa voidaan pitää kriittisenä edellytyksenä onnistuneelle ohjelmistokehitysprojektille (Menezes et al. 2019, s. 1150). Projektinhallinta ja johtaminen ovat jopa tärkeämpiä ominaisuuksia projektin onnistumisen kannalta, kuin sen teknisen toteutuksen yksityiskohtainen hienosäätö (Tesch et al. 2007, s. 61). Emamin et al. (2008, s. 87) suorittamassa kyselytutkimuksessa 28% vastanneista ohjelmistoyrityksistä piti

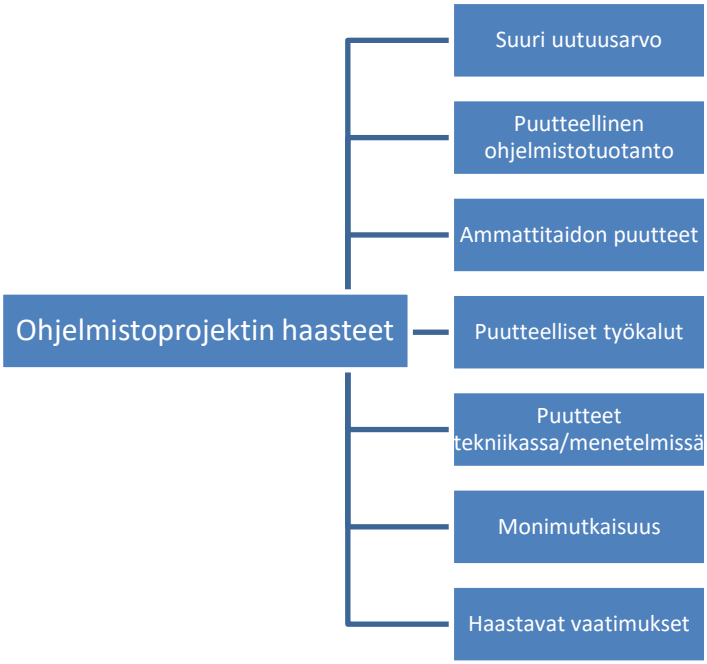
epäonnistunutta projektinhallintaa merkittävimpänä syynä ohjelmistotuotantoprojektiensa epäonnistumiselle. Vastanneet pitivät tätä syytä merkittävämpänä vain huonosti tehtyä vaatimusten määrittelyä ja vaatimusten vaihtelua projektin aikana sekä ylimmän johdon riittämättömyyttä.

Projektinhallintaa ohjelmistoprojekteissa vaikeuttaa myös jatkuvasti muuttuva toimintaympäristö ja alan nopea kehitys. Esimerkiksi Wysocki (2006, s. 1) toteaa jo kymmenessä vuodessa projektien johtamisen ja ohjelmistotekniikan mullistuvan niin, että vanhentuneet projektinhallinnan mallit eivät enää vastaa nykyajan vaatimuksia. Jo yli 15 vuotta sitten Dalcher (2003, s. 421-422) tarkasteli projektinhallinnan ongelmia ohjelmistoalalla ja arvioi kaavoihin kangistuneisuuden sekä lyhytnäköisyyden olevan merkittävä syy projektien epäonnistumisiin ja keskeytyksiin, jotka olisivat olleet estettävissä dynaamisemmalla projektin hahmottamisella ja muutoksiin varautumalla. Usein tutkimusten tuloksena tuotetut ohjelmistoprojektimallit käsittävät hyvinkin tarkasti projektin rakenteen erilaisine vaiheineen, tehtävineen ja prosesseineen. Niissä ei kuitenkaan perehdytä projektin aikatauluttamisen ja kontrolloimisen osa-alueisiin tarkasti, mikä lisää projektinhallinnan haastavuutta ja vaikeuttaa erityisesti johtamista (Ahmed 2011, s. 12).

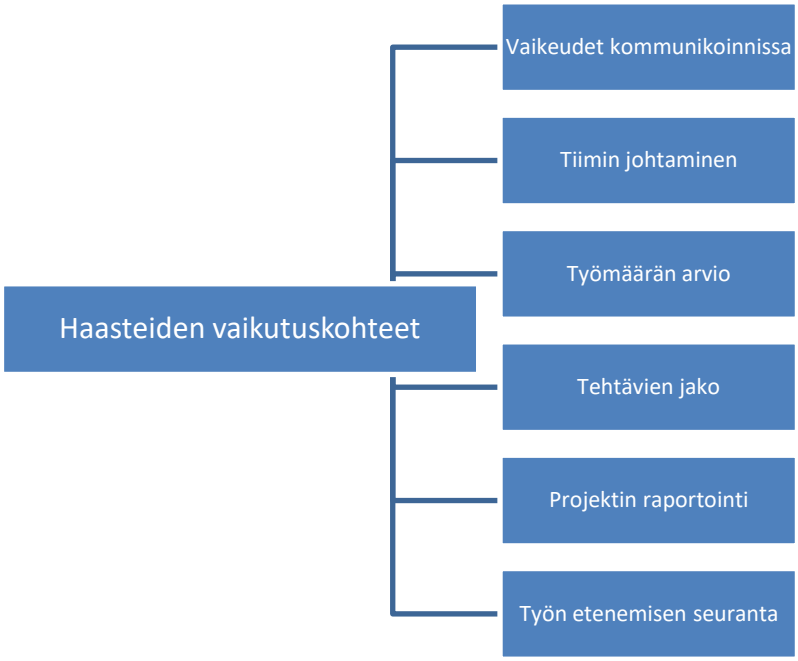
Vaikka aikataulussa ja budjetissa pysyminen ovat tärkeitä kriteerejä onnistuneelle projektille, pitää Blagojevic et al. (2009, s. 191) näihin mittareihin liiallisen keskittymisen kaiken toiminnan ja johtamisen lähtökohtana vähentävän monipuolista ja luovaa suunnittelua. Hänen mukaansa ehdoton keskittyminen mahdollisimman ripeään ja taloudellisesti suotuisaan projektin toteuttamiseen vie huomion pois esimeriksi riskienhallinnalta ja osapuolten vuoropuhelulta. Vaikka aikataulussa ja budjetissa pysyminen ovat varmasti kriittisiä edellytyksiä projektin onnistumiselle, ovat nämä nimenomaan seuraus pienemmistä osa-alueista, kuten onnistuneesta organisaatiokulttuurista, vahvasta osapuolten välisestä vuorovaikutuksesta, selkeästi määritetystä tavoitteesta, selkeästä ohjelmiston määrittelystä ja riskienhallinnasta (Sudhakar 2012, s. 242-243).

Lu et al. (2013) toteaa ohjelmistoprojektien olevan todella laajoja ja sen aiheuttavan haasteita eri tasoilla ja eri lähteistä. Ahmed (2011, s. 8) on jakanut haasteet seitsemään eri kategoriaan. Näitä ovat muun muassa huono määrittelydokumentti, liian suuri monimutkaisuus ja teknologian riittämättömyys. Kaikki seitsemän haastetta on esitetty kuvassa 3.1. Nämä ovat kuitenkin hyvin vahvasti teknologiaan ja sen käyttöön rajoittuvia haasteita, eikä niissä mainita ollenkaan esimerkiksi tiedon jakamiseen ja kommunikointiin liittyviä haasteita, jotka esimerkiksi Menezesin et al. (2019, s. 1158) ja Cinisin (2015) mukaan johtavat usein toteutuneisiin negatiivisiin riskeihin. Sen sijaan teknologiaan ja

ohjelmointiin liittyvät haasteet aiheuttaisivat juurikin nämä kommunikointiin ja johtamiseen liittyviin ongelmiin, kuten on esitetty kuvassa 3.2 (Ahmed 2011, s. 9).



Kuva 3.1: *Ohjelmistoprojektinhallinnan haasteet jaettuna eri kategorioihin (Ahmed 2011, s. 8)*



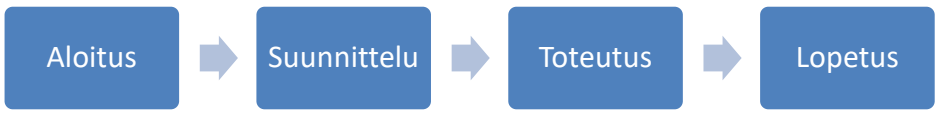
Kuva 3.2: *Osa-alueet, joihin ohjelmistoteknisten haasteet vaikuttavat (Ahmed 2011, s. 9)*

Ohjelmistoprojekteille on myös tyypillistä, että ne eivät käytännössä pääty täysin ikinä, koska vielä projektin toimittamisen ja käyttöönoton jälkeenkin toimitettu ohjelmisto vaatii päivityksiä ja muuta ylläpitoa tai asiakas haluaa siihen kehitettävän lisäominaisuuksia.

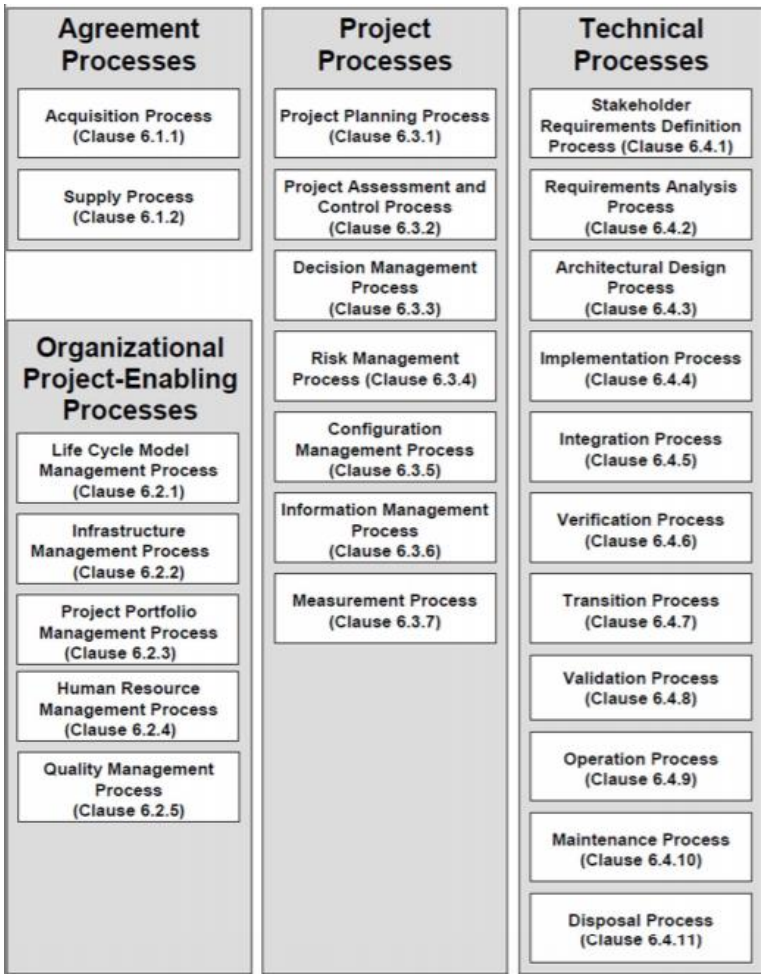
Tämä voi olla ohjelmistoprojekteja toimittavan yrityksen liiketoiminnan jatkuvuuden kannalta positiivinen asia, mutta projektinhallinnan näkökulmasta se aiheuttaa ongelmia projektin onnistumisen arvioinnissa, jos selvää päättymisajankohtaa projektille ei ole havaittu. (Brewer & Dittman 2013)

3.3 Näkökulmia ohjelmistoprojekteihin

Suuresta projektin elinkaaren mittakaavasta tarkasteltuna jokainen projekti koostuu alullepanosta, suunnittelusta, toteutuksesta ja päätöksestä (Medvedska & Berzisa 2015, s. 6). Nämä vaiheet ovat projektinhallinnan suunnittelun lähtökohta. Toisaalta projekteja voidaan tarkastella tiettyjen järjestyksessä olevien vaiheiden sijaan myös prosessien kautta. Esimeriksi Lin & Huang (2018, s. 46) ovat jakaneet ohjelmistoprojektin neljästä erillisestä prosessista koostuvaksi kokonaisuudeksi. Prosessit ovat sopimuksenteko, organisationaaliset prosessit, toteutus ja tekninen prosessi. Nämä useista osaprosesseista koostuvat prosessit muistuttavat kuitenkin osittain Medvedskan & Berzisan (2015) määrittämää vaiheisiin perustuvaa mallia. Esimerkiksi sopimuksenteko on prosessimallissa oma siilonsa, mutta toisaalta se tapahtuu joka tapauksessa projektin alullepanon yhteydessä. Toisaalta erona prosessimallissa on, että moni prosessi on läsnä koko projektin ajan, eikä vain tietyssä vaiheessa. Esimerkiksi projektin toteutukseen kuuluvat tietojohtamisen ja riskienhallinnan osaprosessit kattavat koko projektin elinkaaren, kun taas vaiheena toteutus tarkoittaa ajallisesti toimintaa projektin suunnittelun ja käyttöönoton välissä.



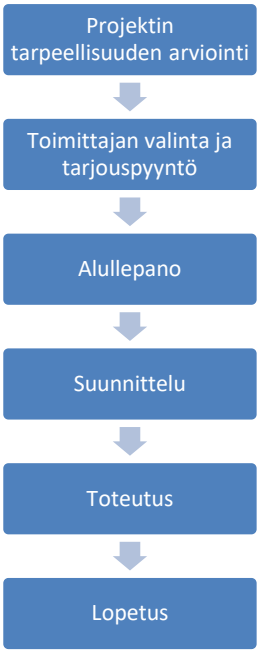
Kuva 3.3: *Yleispätevä pohja projektin vaiheille (Medvedska & Berzisa, 2015)*



Kuva 3.4: *Projektin jako eri prosesseihin (Lin & Huang 2018, s. 46)*

Ohjelmistoprojektin rakenteeseen ja sen hallintaan ei kuitenkaan ole mitään sovittua toimialan normia, vaan aiheeseen on erilaisia näkemyksiä. Chemeturi (2013, s. 19) tiivistää IT-projektin kuuteen vaiheeseen, joista kahdessa ensimmäisessä hän huomioi projektin tilaajan näkökulman. Hän aloittaisi projektin niinkin varhaisesta vaiheesta, kuin projektin tuotoksen käyttökelpoisuuden arvioinnista. Tämän vaiheen lopuksi on tehtävä päätös, aletaanko projektia kehittää pidemmälle vai hylätäänkö se. Olettaen, että projekti tilataan joltakulta ulkopuoliselta ohjelmistotuottajalta, seuraava vaihe on pyytää projektista tarjous ja tehdä tilaus halutulta tuottajalta.

Neljä viimeistä vaihetta noudattavat Medvedskan & Berzisan (2015) neljän vaiheen mallia. Seuraavaksi suoritetaan projektin alullepano, johon kuuluu muun muassa henkilöstön ja roolien valinta, tarvittavien resurssien määrittely ja dokumentoinnin aloittaminen. Neljäntenä vaiheena on yksinkertaisesti projektin toteuttamisen suunnittelu, jota seuraa viidentenä vaiheena projektin toteutus ja kuudentena lopetus.



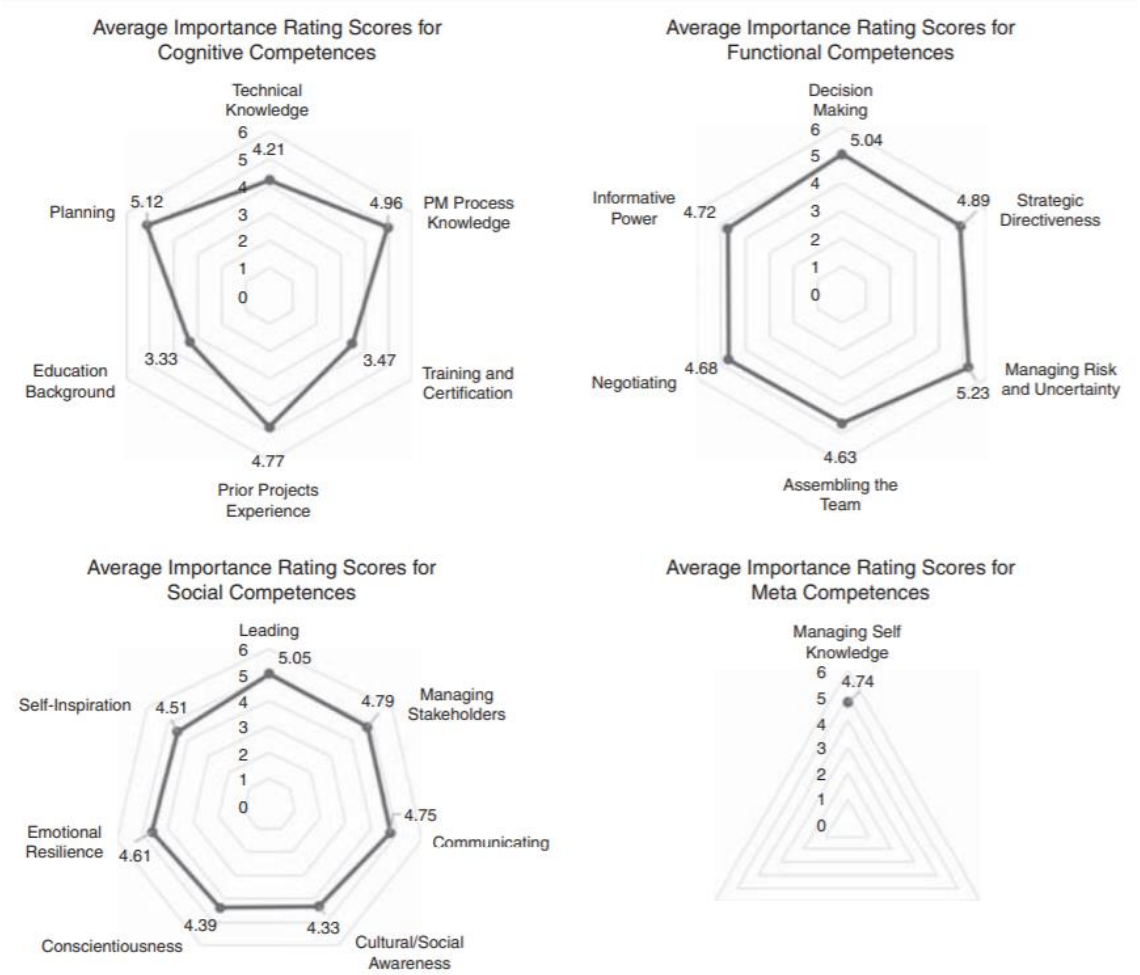
Kuva 3.5: Chemeturin (2013) malli projektin vaiheille

Cha & Maytorena-Sanchez (2019) tarkastelevat projektinhallintaa jakamiensa eri osa-alueiden näkökulmasta. Osa-alueet on jaettu kolmeen eri kategoriaan, jotka ovat kognitiiviset, funktionaaliset ja sosiaaliset taidot. Nämä on edelleen jaettu taitoihin, joista kukin kategoria koostuu. Eri taidot korostuvat eri vaiheissa projektia.

Kognitiiviset taidot ovat tietämykseen ja ymmärryksen kykyjä. Käytännössä tämä tarkoittaa kykyä omaksua ja hyödyntää hiljaista tietoa. Funktionaaliset taidot tarkoittavat kykyä toimia käytännön tasolla tietyssä eli soveltaa teoriassa opittua käytännön tehtäviin. Sosiaaliset taidot ovat kykyjä ymmärtää muita, tulla ymmärretyksi ja toimia erilaisten ihmisten kanssa erilaisissa tilanteissa. Myös neljäs mahdollinen taitokategoria, metaosaaminen eli kyky oppia uutta ja tunnistaa omat vahvuudet ja puutteet, tuodaan ilmi.

Eri kategoriat ja niiden painotus kyselytutkimuksen perusteella esitetään kuvassa 3.2. (Cha & Maytorena-Sanchez 2019, s. 2, 8). Luvut tarkoittavat taitojen tärkeyttä asteikolla yhdestä kuuteen. Cha & Maytorena-Sanchez (2019) saivat luvut haastateltuaan ohjelmistoprojektien johtajia taitojen merkittävyydestä, missä he pyysivät arvioimaan taitojen merkittävyyttä mainitulla asteikolla. Keskimäärin koko projektin elinkaaren aikana kognitiivisista taidoista tärkeimmät ovat suunnittelun ja projektinhallinnan prosessien ymmärtämisen taidot. Funktionaalisissa ja sosiaalisissa taidoissa erot taitojen tärkeydessä ovat

hyvin pienet. Tärkeimpänä funktionaalisena taitona on pidetty päätöksenteon riskienhallinnan taitoa, kun taas sosiaalisista kyvyistä muita selvästi tärkeämpänä pidetään johtamisen taitoa.



Kuva 3.6: *Projektinhallintaan liittyvät taidot jaettuna neljään eri kategoriaan ja niiden koostumukset sekä arvostukset (Cha & Maytorena-Sanchez 2019, s. 8). Tärkeysasteikko on yhdestä kuuteen: mitä korkeampi luku, sitä tärkeämpänä haastatellut projektipäälliköt ovat keskimäärin taitoa pitäneet.*

Tässä luvussa selvitettiin aluksi tutkimukseni keskeisimmät käsitteet. Käsitteiden käytyä tutuksi tarkasteltiin ohjelmistoprojektien haasteista sekä yleisesti eri lähteiden näkemyksiä ohjelmistoprojektien tarkasteluun. Kappaleessa siis luotiin pohja riskienhallinnan näkökulman tarkastelulle.

4. RISKIT OHJELMISTOPROJEKTEISSA

Niin kuin yleisestikin projekteissa, myös ohjelmistoprojekteissa riskienhallinta on tärkeää ja olennainen osa projektinhallintaa. Hyvin toteutettu riskienhallinta vähentää taloudellisia tappioita, koska aikaa ja resursseja säästyy mahdollisten ongelmien korjaamiselta ja niistä palautumiselta (Bedi et al. 2012, s. 1013-1014). Ohjelmistoprojektien riskienhallintaan kiinnitetään silti liian vähän huomiota, vaikka niiden epäonnistuminen on kovin yleistä (Tang et al. 2012, s. 2753). Jotta pystytään tunnistamaan syitä ohjelmistoprojektien epäonnistumisille, on tarkasteltava syvemmin riskejä ja riskienhallinnassa tehtyjä virheitä. Seuraavaksi jaotellaan ohjelmistoprojektien riskejä eri tavoin, sekä tarkastellaan, mitkä niistä ovat yleisimpiä, jotta voidaan ymmärtää paremmin riskit, jotka vaativat eniten huomiota riskienhallinnassa. Lopuksi käydään riskienhallinnan toteuttamiseen liittyviä yleispiirteitä ja haasteita.

4.1 Riskien jaottelu

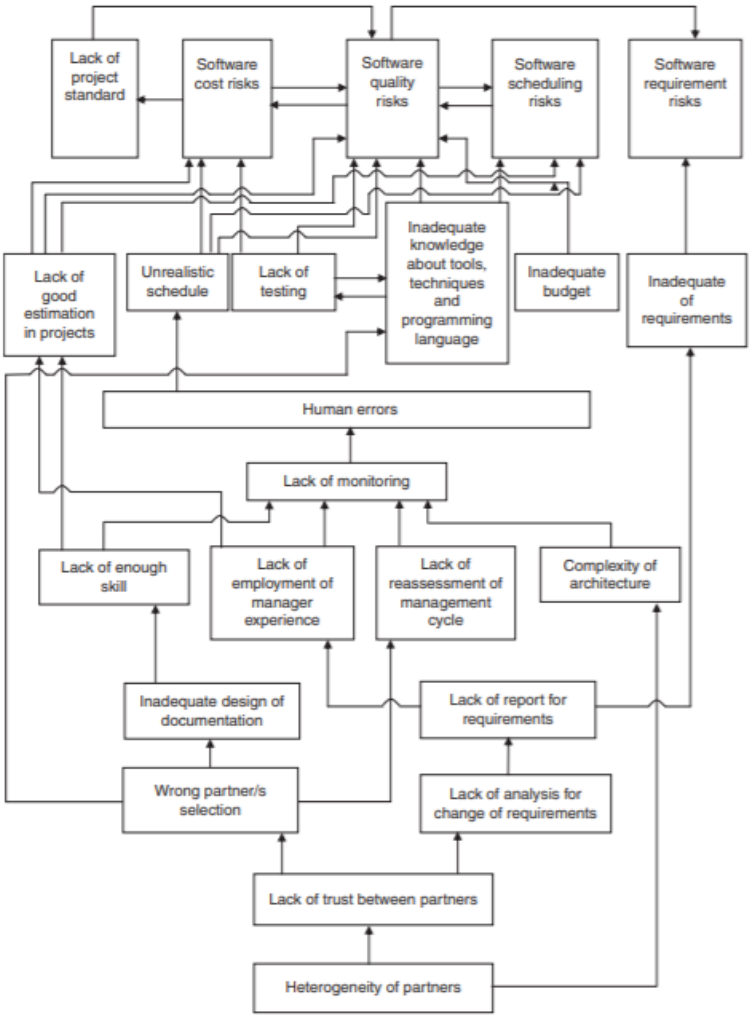
Ohjelmistoprojektien riskejä on tutkittu ja tunnistettu paljon ja niiden jaottelusta on useinta tulkintoja. Lun et al. (2013) mukaan niitä on kymmenen keskeistä. Näitä ovat muun muassa henkilöstön riittämättömyys, epärealistinen budjetti ja aikataulu, väärien ja tarpeettomien toimintojen ohjelmointi, vaatimusten vaihtelu projektin aikana ja jopa tietoteknisten mahdollisuuksien riittämättömyys. Riskejä voidaan jaotella myös yksityiskohtaisemmin, jolloin niitä ymmärrettävästi käy ilmi enemmän, tai vielä suppeammin, jolloin riskit jaetaan ennemminkin aihealueisiin. Esimerkiksi Cinis (2015) pitää ohjelmistoprojektien riskien keskeisinä lähteinä vain paria ”kattokäsitettä”, jotka ovat täsmällisen tavoitteen määrittely ja kaikkien osapuolten sitoutuminen projektin. Samoin McManus (2012, s. 5) on jakanut riskit vain kahteen kategoriaan: liiketoiminnan ja teknologian riskeihin. Liiketoiminnan riskit käsittävät esimeriksi hinnoittelun ja lainsäädännön riskit, kun taas teknologian riskejä ovat muun muassa tuotannon rajoitukset ja käytettyjen teknologioiden patentit. Kategorioiden riskit on esitetty taulukossa 4.1.

Liiketoiminnalliset	Teknologiset
Hinnoittelustrategia	Tuotannon rajoitteet
Lainsäädännölliset rajoitteet	Suunnittelun yhteensopivuus
Kopioinnin riski	Innovointi
Tekijänoikeudelliset maksut	Tarvittavan tekniikan saatavuus
Asiakkaan sitoutuminen	Patentit
Projektin ajankohta	Teknologian ainutlaatuisuus
Sidosryhmien vaikutukset	

Taulukko 4.1: *Riskien jako liiketoiminnallisiin ja teknologisiin riskeihin (McManus 2012, s. 5)*

Sen sijaan Menezes et al. (2019, s. 1158-1159) on tunnistanut ohjelmistoprojekteihin liittyviä riskejä jopa 148 erilaista jaettuna kolmeen eri luokkaan. Kolme pääluokkaa ovat ohjelmistotuotannon, kehitysympäristön ja ohjelmoinnin rajoitteisiin liittyvät riskit. Näihin luokkiin on jaettu riskejä aiheuttavat tekijät. Ohjelmistotuotannon riskejä ovat esimeriksi määrittelyyn ja ohjelmistotestaukseen liittyvät riskit. Kehitysympäristössä riskejä aiheutuu muun muassa johtamisen ja työympäristön ongelmista. Ohjelmointiin yhteydessä olevat riskit ovat esimeriksi tietoteknisestä monimutkaisuudesta ja rajoitteista aiheutuvat riskit. Vaikka Samantra et al. (2016, s. 17) tunnistaa huomattavasti vähemmän riskejä, kuin Menezes et al. (2019), heidän luoma 23 eri riskiä käsittävä kaavio havainnollistaa graafisesti, miten paljon riskejä ohjelmistoprojekteihin liittyy ja miten ne ovat vuorovaikutuksessa keskenään. Kaavio on kuvassa 4.1.

Taylor (2006) jakaa riskit vain neljään pääluokkaan, jotka ovat projektinhallinnan riskit, osapuolten välisiin suhteisiin liittyvät riskit, projektin monipuolisuuden aiheuttamat riskit ja ympäristön aiheuttamat riskit. Projektinhallintaan liittyvä riski voi olla esimeriksi projektipäällikön kokemattomuus (Sauer et al. 2007). Osapuolten välisten suhteiden ja kommunikaation riskeihin kuuluu muun muassa Menezesin et al. (2019, s. 1158) ja Cinisin (2015) havaitsema osapuolten sitoutumattomuus projektiin ja väärinkäsitysten syntyminen esimerkiksi puutteellisen tai virheellisen tiedonvälityksen takia. Projektin monimuotoisuus aiheuttaa riskejä, jos se on liian laaja ja monimutkainen lisäten sen hallitsemisen ja johtamisen haastetta (Martin et al. 2007; Howell et al. 2010). Ympäristön aiheuttamia riski voi olla esimerkiksi projektin haavoittuvuus toimintaympäristön muutoksille (Howell et al. 2010).



Kuva 4.1: *Ohjelmistoprojektien riskit jaoteltuna 23 eri luokaan ja niiden suhteet toisiinsa (Samantra et al. 2016, s. 17).*

Taylor et al. (2012, s. 24-26) jakaa riskit kuuteen eri luokkaan. Tämä jaottelu on monipuolisempi, kuin edellisessä kappaleessa läpikäyty Taylorin kuusi vuotta vanhempi jaottelu. Kuusi riskien ulottuvuutta ovat projektin kriittisyys, epävarmuus, laajuus, monimutkaisuus, projektinhallinta ja sidosryhmien osallistuvuus. Nämä ulottuvuudet käsittelevät osittain samoja riskejä, kuin aiemmat neljä. Lisäksi tarkastellaan riskejä uusista näkökulmista, jotka jäivät huomioimatta aiemmassa tutkimuksessa.

Projektin kriittisyydellä viitataan tässä yhteydessä projektin näkyvyyteen ja julkisuuteen liittyviä riskejä. Esimerkiksi projektin ollessa suuren julkisen huomion kohteena, voi sen osapuolet saada negatiivistä julkisuutta ongelmien ilmetessä projektin aikana (Shenhar 2001). Epävarmuuden ulottuvuudella tarkoitetaan yksinkertaisuudessaan tulevaisuuden ennustamisen mahdottomuutta ja sen näkymistä riskienhallinnassa riskeihin varautumisessa ja vaikutusten arvioinnissa. Projektin monimutkaisuus lisää epävarmuuden riskiä ja usein myös lisää projektin teknisiä vaatimuksia, mikä myös on merkittävä riski projektin

epäonnistumiselle muun muassa Menezesin et al. (2019) ja Lun et al. (2013) mukaan. Projektin laajuuden aiheuttamat riskit osin osittain samoja, kuin monimutkaisuuden. Projektin laajuus myös lisää helposti sen monimutkaisuutta. Laajuus lisää riskiä projektin myöhästymiselle ja huolellisuuden vähenemiselle (Martin et al. 2007; Sauer et al. 2007). Projektinhallinnan riskit liittyvät projektinhallinnan huonoon järjestämiseen ja tehottomaan organisaatiokulttuuriin esimerkiksi projektipäällikön kyvyttömyyden takia. Sidosryhmien osallistumattomuus aiheuttaa riskejä puutteelliselle kommunikoinnille osapuolten välillä ja väärinkäsityksiin, jotka voi johtaa esimeriksi huonosti tehtyyn projektin määrittelyyn ja siihen, etteivät osapuolet ole selvillä projektin vaiheista ja etenemisestä.

4.2 Riskien yleisyys

Riskejä sisältyy koko projektin elinkaaren kaikkiin vaiheisiin (Menezes 2019, s. 1151). Kumar & Rashid (2018, s. 60-63) mainitsee ongelmien ilmenevän kuitenkin vasta projektin loppuvaiheessa, vaikka jo projektin ensimmäiset vaiheet, kuten määrittely ja projektin kokoluokan arviointi, ovat todella riskialttiita ja vaikuttavat koko projektin myöhempiin vaiheisiin. Tang et al. (2012, s. 2753) mukaan lähes 30% ohjelmistoprojekteista perutetaan ennen valmistumista ja 45% projekteista myöhästyy vähintään 175% suunnitellusta aikarajasta. Saman tutkimuksen mukaan vain 25% projekteista valmistuu tavoitellussa ajassa ja budjetissa.

Menezes et al. (2019, s. 1158) 148 tunnistettua riskiä sisältäneen tutkimuksen mukaan kolme yleisintä ongelmia aiheuttanutta tekijää olivat työntekijöiden ammattitaidon riittämättömyys, määrittelyn liiallinen monitulkintaisuus ja asiakkaan puutteellinen osallistuminen projektiin. Kaksi ensimmäistä olivat selvästi yleisimpiä verrattuna seuraaviin, joissa marginaalit yleisyyksien välillä olivat vähäisempiä. Usealla tutkimuksessa listatun kymmenen yleisimmän riskin välillä on havaittavissa selvästi yhteys. Esimerkiksi asiakkaan puutteellinen osallistuminen projektissa haittaa tiedonvälitystä osapuolten välillä, mikä johtanee ympäripyöreisiin määrittelyihin, sekä listalla myös mainittuihin määrittelyiden muutoksiin ja puutteelliseen määrittelyyn. Ympäripyöreä määrittely voi myös lisätä projektin huonon suunnittelun riskiä, joka on myös mainittu listassa, koska ohjelmiston epäselvä määrittely varmasti vaikeuttaa tarkan suunnitelman tekemistä. Täten heikko yhteistyö asiakkaan kanssa on myös riski projektin huonolle suunnittelulle. Työntekijöiden ammattitaidon riittämättömyyden vaikutelma lienee seuraus teknisesti liian monimutkaisista projekteista, joita myös Lu et al. (2013) pitää keskeisimpien ohjelmistoprojektien riskien joukossa.

On huomionarvoista, että useat edellä mainitut riskit ovat olleet yleisimpiä jo yli kymmenen vuotta sitten. Tesch et al. (2007, s. 64) määritti kaksitoista vuotta sitten yleisimmiksi riskeiksi muun muassa henkilöstön taitojen riittämättömyyden ja teknisesti liian vaikean määrittelyn. Riskit ovat kuitenkin olleet yleisiä jo kauan: vuonna 1995 arvioitiin vain 16,2 prosenttia ohjelmistoprojekteista toteutuvan halutun budjetin ja aikataulun rajoissa, 31,1% peruttavan ennen valmistumistaan, sekä 52,7% valmistuvan lähes kolminkertaisella aikataululla suunnitellusta, eli todella vakavasti myöhässä (Samantra et al. 2016, s. 2).

4.3 Riskienhallintaprosessimalleja

Riskienhallinnan tavoite on minimoida projektin negatiivisten riskien toteutuminen ja maksimoida positiivisten riskien tarjoamat mahdollisuudet (Tesch et al. 2007, s. 62). Ohjelmistoprojektien riskienhallintaan on alettu kiinnittää huomiota, koska monen tärkeän projektin epäonnistuminen huonon riskienhallinnan takia on aiheuttanut suuria taloudellisia tappioita (Tang et al. 2012, s. 2753).

Riskienhallintaprosessi sisältää useita osaprosesseja. Riskienhallinnan suunnittelu on riskienhallintatoimintojen suunnittelun ja toteuttamisen päättämistä. Riskien tunnistaminen sisältää projektiin vaikuttavien riskien arvioinnin ja niiden tunnuspiirteiden dokumentoinnin. Tunnistettujen riskien analysointi on lähinnä riskien priorisointia ja niiden toteutumisen todennäköisyyden ja vaikutusten arviointia. Riskien vaikutusten numeerisessa analysoinnissa riskien vaikutus projektin tavoitteisiin arvioidaan tunnusluvuin. (Tesch et al. 2007)

Myös Tang et al. (2012, s. 2754) painottaa riskien tunnistamisvaihetta. Se tulisi kuitenkin purkaa mahdollisimman täsmällisiin osiin. Hänen mukaan Teschin et al. (2007) mainitseman seurausten arvioinnin kohteena tulisi olla esimerkiksi rahalliset, aikataululliset ja teknilliset seuraukset.

Näiden toimintojen jälkeen voi suorittaa kaksi edellisten vaiheiden tuottamien tietojen mahdollistamaa prosessia. Riskeihin reagoimisen suunnittelu on negatiivisten riskien tuottamiin uhkiin ja positiivisten riskien tuottamiin mahdollisuuksiin varautumista. Erityisesti keskittyminen on negatiivisten riskien välttämisessä tai toteutuessa niiden vaikutusten mitätöimisessä. Riskienseurantaa toteutetaan koko projektin ajan sisältäen uusien riskien tunnistamista, nykyisten riskien toteutumisen arviointia ja seurantaa, sekä toiminnan kehittämistä. (Tesch et al. 2007, s. 62)

Myös Menezes et al. (2019) pitää riskien tunnistamista ja monitorointia ratkaisevana projektin onnistumiseksi. Käytännön tasolla tämä ei kuitenkaan toteudu, vaan näiden menetelmien puutteellinen suorittaminen on yksi keskeisimpiä syitä ohjelmistoprojektien epäonnistumiseksi. Jo yli kymmenen vuotta sitten lähes kolmasosa ohjelmistoprojekteista epäonnistui juurikin puutteellisen riskienhallinnan vuoksi (Emam et al. 2008).

Useassa riskienhallintaa käsittelevässä tutkimuksessa vaikuttaa ilmenevän sama kaava ideaaliselle riskienhallintaprosessille. Myös Taylor et al. (2012), joka tutkimuksessaan on kerännyt tutkimuksia ohjelmistoprojektien riskeistä yli 30 vuoden ajalta, pitää teoriassa parhaimpana mallina prosessin aloittamista riskien tunnistamisesta, seuraavana toteuttaen riskien todennäköisyyksien sekä seurausten analysoinnin ja priorisoinnin. Kolmantena tulee luoda suunnitelma riskeihin reagoimiseen niiden toteutuessa. Näiden vaiheiden oltua tehty huolellisesti voi keksittyä riskien seurantaan koko projektin ajan.

Edellisistä poiketen Bedi et al. (2012, s. 1014) määrittäisi kolmivaiheisessa mallissaan riskien tunnistamisen jälkeen vastuuhenkilöiden määrittämisen riskeille osa-aluiden mukaan. Näiden vastuuhenkilöiden- tai tiimien tehtävä olisi valvoa omaan vastuualueeseen kuuluvien riskien toteutumisen mahdollisuutta ja varautua niihin etukäteen mahdollisimman hyvin. Tämä lienee järkevää hyvin isoissa ja mittavissa projekteissa, joissa riskienhallinta on todella laaja toiminto. Pienempiin osiin jakaminen vähentää yhden ison riskienhallintayksikön taakkaa ja pienemmät yksiköt voivat keskittää resurssinsa täysin omaan vastuualueeseensa.

4.4 Riskienhallinnan haasteita

Vaikka ohjelmistoprojektien riskejä ja riskienhallintaa on tutkittu yli 30 vuoden ajan, käytännön tasolla tutkimusten tuottamaa tietoa ja käytäntöjä riskienhallinnan tueksi ei hyödynnetä (Taylor et al. 2012, s. 2753). Yksi merkittävä syy hänen mukaan tähän voi olla tutkimusten lähtökohta siitä, että projektipäällikkö tai muu riskienhallinnasta vastaava tietäisi kaikki projektin riskit, niiden todennäköisyydet sekä niiden vaikutukset. Itse asiassa hyvissä ajoin tunnistettuihin riskeihin reagoimiseen on luotu hyödyllisiä käytäntöjä ja toteutuessa niiden aiheuttamat vahingot ovat verrattain pieniä (Bedi et al. 2012, s. 1013, 1016).

Todellisuudessa riskienhallinta ja riskien tunnistaminen ei kuitenkaan ole yksinkertaista saatikka helppoa, vaan mahdollisia riskejä jää aina tunnistamatta. Tämä johtunee projektien monimuotoisuudesta sekä siitä, että niihin vaikuttaa useita sidosryhmiä ja muuttujia koko projektin ajan (Lu et al. 2013). Myöskään riskien toteutumisen todennäköisyyksistä tehdyt arviot ja toteutumisen seuraukset ovat vain inhimillisiä arvioita, eivät faktoja, mitä ei välttämättä huomioida riskienhallinnan tutkimuksissa (Taylor et al. 2012, s. 2753).

Tehokas riskienhallinta on proaktiivista eli riskeihin etukäteen varautuvaa, eikä reaktiivista, eli riskeihin reagoimista ensimmäisen kerran vasta sitten, kun ne ovat jo toteutuneet. Tätä voi havainnollistaa arkielämän esimerkillä paloturvallisuudesta. Tulipaloja aiheutunee harvoin, jos talon asukkaat ovat tietoisia paloturvallisuusohjeista ja -käytänteistä tulipalojen välttämiseksi. Tulipalon sattuessa aiheutunee huomattavasti vähemmän vahinkoja, jos talossa on muun muassa palosammutin ja sammutuspeite, joiden käyttöön henkilöstö on koulutettu. Vaikka esimerkki kuulostanee itsestään selvältä, ei sen logiikan toteuttaminen ole kuitenkaan niin yksinkertaista ohjelmistoprojektien riskienhallinnassa.

Fatima (2017, s. 10) pitää proaktiivisen riskienhallinnan puuttumista yhtenä ohjelmistoprojektien epäonnistumisen keskeisenä syynä proaktiivisen riskienhallinnan puutetta. Tätä väitettä tukee Taylorin et al. (2012, s. 2753) toteamus ohjelmistoprojektien riskien vaikeasta ennustamisesta, jota lisää Lun et al. (2013) huomio projektien laajuudesta ja sen myötä monimutkaisuudesta. Blagojevic et al. (2009, s. 191) väittää riskienhallinnan lähtökohdan ohjelmistokehityksessä luottavan liikaa siihen, että ohjelmisto on helposti eri tilanteisiin mukautuva ja jälkeenpäin muokattavissa, minkä seurauksena proaktiivisten menetelmien luomista ja käyttämistä ei kehitetä tarpeeksi.

Riskejä siis voi jakaa usealla tavalla. Lisäksi riskienhallinnasta vastaava päättää, kuinka yksityiskohtaisesti riskit jaetaan, koska niitä voi tulkita olevan muutamia, kymmeniä tai jopa satoja. Tutkimustulosten mukaan riskien yleisyys ei ole muuttunut tällä vuosituhannella merkittävästi, vaikka ala on kasvanut ja teknologia kehittynyt. Riskienhallinta on kuitenkin kriittisen tärkeä toiminto, jota ei kuitenkaan toteuteta riittävän hyvin. Tämä voi johtua useista haasteista, joita riskienhallinnan käytännön toteutukseen liittyy.

5. OHJELMISTOPROJEKTIEN EPÄONNISTUMISTEN ENNALTAEHKÄISY RISKIENHALLINNALLA

Tässä kappaleessa teen päätelmät tutkimuksen aiheesta kahden edellisen kappaleen tietojen perusteella. Keskityn aluksi pohtimaan toimivan riskienhallinnan mahdollisuuksia projektinhallinnan haasteiden minimoimiseksi ja erityisesti projektin epäonnistumisen ehkäisemiseksi. Lopuksi pohdin lyhyesti, mitä haasteita riskienhallinnan yhteensovittamisella projektinhallinnan menetelmiin voi olla.

5.1 Riskienhallinnan hyödyntäminen projektinhallinnassa epäonnistumisten ehkäisemiseksi ja haasteiden minimoimiseksi

Riskienhallinnallalla on merkittävä vaikutus projektin onnistumiselle. Onnistuneella riskienhallinnalla ehkäistään negatiivisia riskejä (Tesch et al. 2007), tunnistetaan niitä ja varaudutaan niihin etukäteen (Menezes et al. 2019) ja osataan toimia oikein niiden toteutuessa (Taylor et al. 2012). Muun muassa Bannerman (2008, s. 2120) on tehnyt jopa kokonaisen tutkimuksen riskienhallinnalla saavutettavista hyödyistä ohjelmistoprojekteissa. Nämä hyödyt hänen mukaan ovat ideaalisten toimintatapojen tunnistaminen, luottamuksen kasvu projektin onnistumiselle, suurempi todennäköisyys projektin onnistumiselle, vähemmät yllätykset projektin aikana, epävarmuuden väheneminen ja jopa vähempi työpanos koko projektille, koska päällekäisiltä ja turhilta työtehtäviltä voidaan välttyä.

Yhtenä merkittävänä ongelmana ohjelmistoprojekteissa on pidetty niiden laajuutta, joka aiheuttaa projektinhallinnan näkökulmasta haasteita tehtävien jakamisessa ja aiheuttaa helposti vääriin asioihin keskittymistä (Ahmed 2011). Hyvin toteutetulla riskinhallinnalla voidaan tunnistaa jo varhaisessa vaiheessa osa-alueita, joilla on suurin vaikutus projektiin (McManus 2012, s. 6). Riskienhallinnan avulla projektipäällikkö voi siis tietää projektiin vaikuttavat kriittisimmät vaiheet, sidosryhmät ja prosessit, mikä auttanee resurssien kohdistamista oikeisiin tehtäviin.

Cha & Maytorena-Sanchez (2019, s. 8) pitävät tärkeinä projektinhallinnan kykyinä johtamisen, kommunikoinnin ja erilaisten kulttuurien ymmärtämisen taitoja. Tällaiset sosiaaliset taidot voivat ehkäistä Menezesin (2019, s. 1158) ja Cinisin (2015) yleisenä pitämiä

huonon kommunikaation ja puutteellisen tiedonvälityksen aiheuttamia riskejä, joita ovat esimerkiksi huolimattomasti tehty määrittelydokumentti (Menezes 2019) ja määrittelyn muuttuminen projektin aikana (Lu et al. 2013). Parempi kommunikointi projektin tuottajan ja tilaajan välillä voisi myös auttaa asiakasta ymmärtämään, millaisen ohjelmiston hän oikeasti tarvitsee ja mikä on teknisesti mahdollista tuottaa. Tämä ehkäisee riskiä teknisesti liian vaativista ja laajoista projekteista, joissa saatetaan tuottaa ominaisuuksia, joita asiakas ei tiedostamattaan edes tarvitse. Lisäksi teknisen ymmärryksen luominen vähentäisi Blagojevicin et al. (2009, s. 191) toteamaa riskiä liiallisesta luottamisesta ohjelman muokattavuuteen ja mukautuvuuteen ongelmien sattuessa. Täten tiedon välitys ja kommunikointi edistävät proaktiivista riskienhallintaa reaktiiviseen malliin luottamisen sijaan.

Onnistunut kommunikointi ja yhteisymmärrys osapuolten välillä ehkäisevät myös ohjelmistoprojektien selvästi kahta tyypillisintä ongelmaa: myöhästymistä ja budjetin ylittämistä. Jos ohjelmiston tilaava asiakas onnistuneesti selittää toimittajalle, mihin ongelmaan ohjelmistoa tarvitaan ja mitä siltä halutaan, toimittaja ymmärtää millaisia ohjelmistoteknisiä vaatimuksia projektille on ja osaa suunnitella ohjelmistotuotannon näkökulmasta tarvittavat toimenpiteet ja resurssit. Vastaavasti toimittaja voi selittää asiakkaalle oman näkökulmansa vaatimuksista asiakkaan esittämään ongelmaan ja perustella, miksi jokin asiakkaan toivoma ominaisuus voi olla mahdoton toteuttaa tai jopa ongelman ratkaisemisen kannalta turha.

Tämän vuoropuhelun pohjalta voidaan laatia määrittelydokumentti, joka ei sisällä turhia vaatimuksia ohjelmistolle. Turhien ominaisuuksien kehittäminen säästää sekä aikaa, että resursseja. Toimiva kommunikointi auttaa täsmällisen määrittelydokumentin luomista. Tämä vähentää monitulkintaisen määrittelyn aiheuttamia riskejä ja takaa, että määrittely on molempien osapuolten mielestä paikkansapitävä projektille, eikä sitä tarvitse muuttaa myöhemmin projektin aikana. Tämä taas vähentää yllättäviä muutoksia projektissa ja epävarmuutta, mikä helpottaa ja selkeyttää projektinhallintaa.

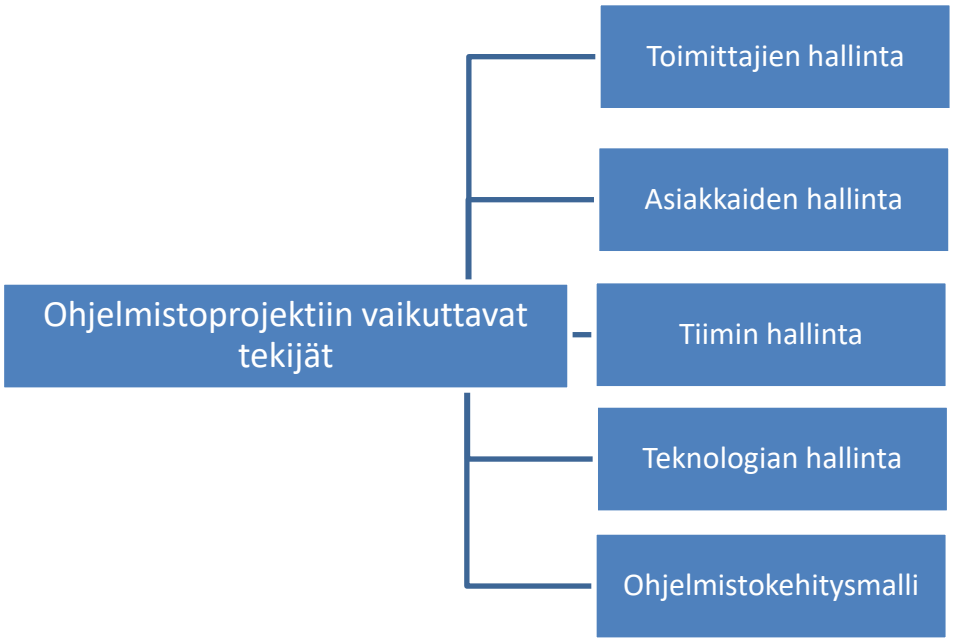
Projektien laajuuden ja monimuotoisuuden aiheuttamat riskit voidaan minimoida jakamalla riskienhallinnan vastuut pienenpiin osiin usealle vastuuhenkilölle eri osastojen mukaan, mitä esimerkiksi Bedi et al. (2012) pitää välttämättömänä toimintatapana riskien tunnistamisen jälkeen. Tätä menettelyä voisi toisaalta hyödyntää jo riskientunnistamis- ja arvioimisvaiheessa, koska muun muassa Bannermanin (2008, s. 2125) mukaan havaituista riskeistä juuri kriittisimpien erottaminen on projektipäälliköille vaikeinta riskien-

hallinnassa. Jo riskien tunnistamisvaiheessa jokainen sidosryhmäänsä erikoistunut vastuuhenkilö voisi hyödyntää oman asiantuntevuutensa omalta osa-alueeltaan, jotta edellä mainitulta haasteelta vältyttäisiin.

Tässäkin mallissa korostuu kommunikaation ja tiedonvälityksen tärkeys. Erityisesti projektipäällikön tulee tietää eri osa-alueisiin ja sidosryhmiin vaikuttavat riskit, jotta hän osaa suunnitella projektin toteutuksen riskit huomioiden ja kohdentaa resursseja riskienhallinnan näkökulmasta kriittisimmiksi havaittuihin vaiheisiin ja prosesseihin. Tällöin ehkäistään yllättävien riskien syntyminen, koska ne on jo tunnistettu kattavasti etukäteen ja niihin on osattu varautua hyvissä ajoin.

5.2 Haasteet yhteensovittaessa riskienhallinnan menetelmiä projektinhallintaan

Toisaalta on havaittu, että sekä projekti- että riskienhallintamalleja on useanlaisia, joten haasteena onkin sovittaa erilaiset riskienhallinnan mallit ja tulkinnat monimuotoiseen projektii. Alla oleva kuva 5.1 havainnollistaa, miten laaja-alaisesti projektiin vaikuttavia sidosryhmiä on. Haasteena voi olla yhdistää sekä kunkin sidosryhmän, että ohjelmistokehityksen asiantuntevuus.



Kuva 5.1: *Yksi tulkinta ohjelmistoprojektiin vaikuttavista tekijöistä (Ahmed 2011, s. 11)*

Riskienhallintamenetelmien yhteensovittaminen projektinhallinnan menetelmiin on myös haastavaa siksi, että riskienhallintaa ei ole jaettu yhteneväisiin vaiheisiin projektinhallintamenetelmien kanssa. Esimerkiksi kuvan 4.1 mallissa riskit on jaettu hyvin selkeästi kahteen eri kategoriaan, mutta niiden vaikutukseen ja ilmenemiseen projektin vaiheissa tai prosesseissa ei jaon tehnyt McManus (2012) ota kantaa. Kuvasta 4.2 käy ilmi, että riskejä on todella jaettu hyvinkin tarkasti erilaisiin kategorioihin ja on jopa tulkittu niiden vuorovaikutussuhteita. Tämä on tietysti hyödyllistä tietoa projektipäällikölle, mutta siitäkään ei käy suoraan ilmi riskien suhdetta projektin vaiheisiin ja prosesseihin.

6. YHTEENVETO

Tutkin ohjelmistoprojektien epäonnistumisen syitä ja riskejä, sekä etsin keinoja välttää projektin epäonnistuminen tai hylkääminen. Etsin keinoja riskienhallinnan näkökulmasta. Tällaisten keinojen tunnistaminen säästäisi varmasti rahaa ja aikaa sekä parantaisi ohjelmistoyritysten asiakastyytyväisyyttä, koska viallista tuotetta ei tarvitsi ottaa käyttöön vain huomatakseen sen vaativan korjausta. Näkökulmani tutkimukseen oli siis projektinhallinnallinen, eikä niinkään ohjelmistotekninen

Etsin tieteellisistä julkaisuista tietoa projektinhallinnasta yleisesti ohjelmistoalalla sekä siinä havaittuja haasteita. Vertailin myös erilaisia näkemyksiä ja havaitsin paljon yhtäläisyyksiä erilaisissa tulkinnoissa ohjelmistoprojektien olemuksesta. Projektit voidaan jakaa joko vaiheiden tai prosessien mukaan, mutta nämä eivät sulje toisiaan pois. Oikeastaan nämä tulkinnat voivat jopa tukea toisiaan tunnistamalla projektia suunniteltaessa eri prosessien merkitys eri vaiheissa.

Tarkastelemassani tutkimusaineistossa kävi ilmi, että selvästi suurin haaste projektinhallinnassa on projektien laajuus, mikä vaikeuttaa suunnittelua, kommunikointia ja johtamista. Tämä taas vaikeuttaa kriittisten tehtävien tunnistamista ja resurssien kohdistamista niihin, mikä taas johtaa yllättäviin muutoksiin ja tunnistamattomien riskien toteutumiseen.

Riskienhallinnan ongelmaksi havaitsin sen teorian tutkimusten tulosten soveltamattomuuden käytäntöön. Ongelma johtuu joko siitä, että jokainen projekti on niin uniikki ja laaja, ettei ainakaan toistaiseksi ole edes ollut mahdollista keksiä yleispätevää riskienhallintamallia. Toinen vaihtoehto on, ettei riskienhallinnan mahdollisuuksia tiedetä ja siten siihen ei käytetä riittävästi resursseja, mikä johtaa huonoon riskienhallintaan ja siten toteutuneisiin riskeihin, joihin ei ole varauduttu riittävästi tai joita ei ole edes tunnistettu etukäteen. Tämä taas johtaa pahimmillaan projektin epäonnistuneeseen lopputuotteeseen tai koko projektin keskeytykseen.

Perustavanlaatuiseksi ongelmaksi havaitsin lopulta puutteellisen kommunikoinnin ja tiedonvälityksen projektin osapuolten välillä. Jo alussa laiminlyöty vuoropuhelu johtaa väärinkäsityksiin projektin vaatimuksista, mikä aiheuttaa ylimääräistä työtä ja kuluttaa resursseja turhaan. Liian tulkinnanvarainen määrittelydokumentti myös lisää epävarmuutta

koko projektin ajalle aiheuttaa riskin yllättäviin muutoksiin, koska heti alusta asti ei ole määritelty tarkasti ja ytimekkäästi mitä projektilta halutaan. Nämä ongelmat aiheuttavat yleisimmät projektien epäonnistumiseksi laskettavat kriteerit: projektin myöhästymisen ja budjetin ylittymisen.

Tämän takia riskienhallinnan tehokas hyödyntäminen alusta asti ehkäisee merkittävästi projektin epäonnistumisen mahdollisuutta. Se vaatii asiantuntijoita arvioimaan kuhunkin sidosryhmään ja prosessiin liittyviä riskejä ja raportoimaan niistä projektipäällikölle. Tällöin projektipäällikkö pystyy huomioimaan kriittisimmät riskit koko projektin elinkaaren ajan päätöksenteossa ja toimintaa suunnitellessa.

Vaikka ohjelmistotuotantoa ja siinä toteutuvia epäkohtia on merkittävästi tutkittu jo viime vuosituhannen vaihteen ajoista asti, projekteja epäonnistuu ja niissä tehdään merkittäviä virheitä vielä nykyäänkin. Ala on kasvanut mullistavasti vuosituhannen vaihtumisen ajoista, sekä teknologia ja tekniikka ovat kehittyneet huomattavasti. Voidaan siis todeta, että pelkkä riskien ja riskienhallinnan soveltamisen tutkiminen eivät riitä ratkaisemaan kaikkia ongelmia. Aihepiiriin voisi tehdä jatkotutkimusta esimerkiksi tutkimuskysymyksellä ”Miten ohjelmistoprojektit onnistuisivat paremmin?”.

LÄHTEET

Ahmed, A 2011, Software Project Management: A Process-Driven Approach, Auerbach Publishers, Incorporated, London.

Aladwani, A. M. 2002, ”IT project uncertainty, planning and success: An empirical investigation from Kuwait.", *Information Technology & People* 15.3: 210-226.

Bannerman, P.L. 2008, "Risk and risk management in software projects: A reassessment", *The Journal of Systems & Software,* vol. 81, no. 12, pp. 2118-2133.

Bedi, P., Gandotra, V., Singhal, A., Narang, H. & Sharma, S. 2012, "Threat-oriented security framework in risk management using multiagent system: PROACTIVE RISK MANAGEMENT", *Software: Practice and Experience,* vol. 43, no. 9, pp. 1013-1038.

Blagojevic, V., Codenie, W., Dedecker, J., Gonzalez-Deleito, N., Deleu, J. & Boucart, N. 2009, "Murphy: A Web 2.0 approach for proactive risk management in hardware/software co-design", IEEE, , pp. 191.

Brewer, J. L. and Dittman, K. C. (2013) *Methods of IT Project Management: Second Edition*. West Lafayette, Ind: Purdue University Press. Available at: http://search.ebscohost.com/login.aspx?direct=true&AuthType=cookie,ip,uid&db=nlebk&AN=599406&site=ehost-live&scope=site (Accessed: 8 October 2019)

Butterfield, A., Ngondi, G., & Kerr, A. 2016, (Eds.), A Dictionary of Computer Science.: Oxford University Press.

Cha, J. & Maytorena-Sanchez, E. 2019, "Prioritising project management competences across the software project life cycle", *International Journal of Managing Projects in Business*

Chang, C. K., Jiang, H., Di, Y., Zhu, D., Ge, Y. 2008, ”Time-line based model for software project scheduling with genetic algorithms, Information and Software Technology”, Volume 50, pp. 1142-1154

Chemuturi, M. 2013, Mastering IT Project Management: Best Practices, Tools and Techniques, J. Ross Publishing, Plantation. Available from: ProQuest Ebook Central.

Cinis, H. 2018, "Improving the Definition of Software Development Projects Through Design Thinking Led Collaboration Workshops", ACM, pp. 254.

Coleman, G. 2016, "Agile Software Development", *Software Quality Professional,* vol. 19, no. 1, pp. 23-29.

Dalcher, D. 2003, "Beyond Normal Failures: Dynamic Management of Software Projects", *Technology Analysis & Strategic Management,* vol. 15, no. 4, pp. 421-439.

Dionne, G. 2013, "Risk Management: History, Definition, and Critique", *Risk Management and Insurance Review,* vol. 16, no. 2, pp. 147-166.

Emam, K. E., & Koru, A. G. (2008). A replicated survey of IT software project failures. IEEE Software, 25(5), 84-90.

Fatima, T. 2017, *A Predictive Analytics Approach to Project Management: Reducing Project Failures in Web & Software Development Projects*, The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), Athens.

Howell, D., Windahl, C. and Seidel, R. (2010). A Project Contingency Framework Based on Uncertainty and its Consequences, International Journal of Project Management 28(3): 256–264.

Kriikku, M. 2019, “Hätäkeskuslaitoksen uusi järjestelmä ajoittain pois käytöstä – ‘Kyllä siinä on henki vaarassa aika usealla henkilöllä Suomessa’” Saatavilla verkossa: https://yle.fi/uutiset/3-10973821 (luettu 14.10.2019).

Kumar, M. & Rashid, E. 2018, "An Efficient Software Development Life Cycle Model for Developing Software Project", International Journal of Education and Management Engineering, vol. 8, no. 6, pp. 59.

Lenberg, P., Feldt, R. & Wallgren, L. S. 2015, ”Behavioral software engineering: A definition and systematic literature review, Journal of Systems and Software”, vol. 107: pp. 15-37.

Lin, Y. & Huang, S. 2018, "The Design of a Software Engineering Lifecycle Process for Big Data Projects", *IT Professional,* vol. 20, no. 1, pp. 45-52.

Lu, S., Yu, S., Chang, D., & Su, S. (2013). Using the fuzzy linguistic preference relation approach for assessing the importance of risk factors in a software development project. *Mathematical Problems in Engineering, 2013* doi: http://dx.doi.org.libproxy.tuni.fi/10.1155/2013/376375

Martin, N.L., Pearson, J.M. and Furumo, K. (2007). IS Project Management: Size, practices and the project management office, Journal of Computer Information Systems 47(4): 52–60.

McManus, J. (2012). *Risk management in software development projects*. Routledge.

Medvedska, O. & Berzisa, S. 2015, "Selection of Software Development Project Lifecycle Model in Government Institution", *Information Technology and Management Science,* vol. 18, no. 1, pp. 5-11.

Menezes, J., Gusmão, C. & Moura, H. Software Qual J (2019), “Risk factors in software development projects: a systematic literature review”, vol. 27: pp 1149-1174.

Pressman, R. S. (2005). *Software engineering: a practitioner’s approach*. Palgrave Macmillan.

Samantra, C., Datta, S., Mahapatra, S.S. & Debata, B.R. 2016, "Interpretive structural modelling of critical risk factors in software engineering project", *Benchmarking,* vol. 23, no. 1, pp. 2-24.

Sauer, C., Gemino, A. and Reich, B.H. (2007). The Impact of Size and Volatility on IT Project Performance, Communications of the ACM 50(11): 79–84.

Shenhar, A.J. (2001). One Size Does Not Fit All Projects: Exploring classical contingency domains, Management Science 47(3): 394–414.

Sommerville, I. 2006. *Software engineering*. 8. ed. Harlow: Addison-Wesley.

Sudhakar, G.P. 2012, "A model of critical success factors for software projects", *Journal of Enterprise Information Management,* vol. 25, no. 6, pp. 537-558.

Tang, X.Y., Jiang, Y.N. & Jian, J. 2012, "Fuzzy Comprehensive Evaluation for IT Project Risk Management", *Applied Mechanics and Materials,* vol. 229-231, pp. 2753.

Taylor, H., Artman, E. & Woelfer, J.P. 2012, "Information technology project risk management: bridging the gap between research and practice", *Journal of Information Technology,* vol. 27, no. 1, pp. 17-34.

Taylor, H. (2006). Risk Management and Problem Resolution Strategies for IT Projects: Prescription and practice, Project Management Journal 37(5): 49–63.

Tesch, D., Kloppenborg, T. J., & Frolick, M. N. (2007). IT PROJECT RISK FACTORS: THE PROJECT MANAGEMENT PROFESSIONALS PERSPECTIVE. *The Journal of Computer Information Systems, 47*(4), 61-69.

Vaismaa, K. 2009, ”Aiheesta analyysiin – Tukipaketti kandidaatin- ja diplomityön tutkimusprosessiin”. Tampereen teknillinen yliopisto, *Tiedonhallinnan ja logistiikan laitos*

Valtiovarainministeriö 2017. ”Ohje riskienhallintaan” Saatavilla verkossa: http://julkaisut.valtioneuvosto.fi/bitstream/handle/10024/80013/VM_22_2017.pdf

Vanhala L. 2012, ”Näin VR sotki lippujärjestelmänsä – Miksi it-projektit epäonnistuvat?” Saatavilla verkossa: https://suomenkuvalehti.fi/jutut/kotimaa/nain-vr-sotki-lippujarjestelmansa-miksi-it-projektit-epaonnistuvat/ (luettu 14.10.2019).

Wysocki, R. K. (2006) *Effective Software Project Management*. Indianapolis, IN: Wiley.